Towards Scalable Security: On the Scalability of Security in Large-Scale Software Systems

Michael A. Enescu
Vancouver, Canada
menescu@interchange.ubc.ca

ABSTRACT

As networked computer systems become ever more pervasive, so too does the field of computer security. Very large-scale networked systems are now commonplace, and these systems demand protection from adversaries. However, there is a fundamental friction between an increase in system complexity and the effort, resources, and techniques required to protect the system. This survey presents an overview of the major issues relating to the feasibility and scalability of employing security techniques in software systems of increasing scale. In doing so, the body of research in related fields is synthesized into a cohesive whole and some conclusions are drawn regarding theoretical scalability of security as a paradigm.

Categories and Subject Descriptors

C.2.0 [**General**]: Computer Security

General Terms

Security, Scalability

Keywords

Scalable Security

1. INTRODUCTION AND BACKGROUND

There has been much research effort expended to study, at least indirectly, the scalability of the various subfields under the umbrella of computer security. As a first step, it would be prudent to present a definition of the term *scalable*. However, it turns out the term is nuanced and non-trivial to define, and we[1] defer a workable discussion on its definition to §1.2. Looking ahead, the definition will be a definition-by-contexts; *scalability* will refer to different things in relation to different security issues (contexts) throughout the paper. Therefore it is advisable to first list the general issues that will be discussed, and a definition of *scalability* by contexts can follow.

[1]Although there is only one author, this paper will use the plural pronoun as is customary in the literature

1.1 Topics to be Discussed

Here we list the general topics for which underlying issues will be discussed in a security-scalability context, following the introduction section:

1. Generic software failures (bugs leading to security vulnerabilities)
2. Intrusion detection
3. Cryptographic algorithms, and associated network protocols
4. Distributed systems
5. Data Storage

1.2 Defining Scalability

The term *scalability* and its use in the context of this paper must now be explicated in order to make more clear the scope of the paper. The difficulty of formally defining *scalability* has been noted for decades in the literature [21], and often represents a marketing buzzword more than a formal property of a system. Although there have been numerous definitions in specific contexts, such as parallel processing scalability [37, 30, 8], these special cases are inadequate for a general definition of the term. A formal and precise definition of scalability is beyond the scope of this paper, but informally a definition by parts (contexts) of scalability follows.

1. In the context of cryptography, scalability will be taken to include computational complexity with respect to algorithmic techniques. It will also include communication costs (network latency) when dealing with associated cryptographic protocols such as those employed in key exchange.
2. From a software engineering perspective, the term will be taken to mean the marginal increase in effort and skill required to construct secure, vulnerability-free applications of large scope. That is, it will represent the feasibility of creating extremely large systems while maintaining acceptable levels of quality (such as freedom from bugs).

3. When considering a large networked system as a whole, scalability refers to the ability to efficiently handle a large number of network nodes and large volumes of data and network traffic. Scalability of data storage is defined analogously.

4. From the view of intrusion detection systems, scalability will mean the ability to correctly classify a large volume of network traffic in a timely manner, while keeping error and false alarm rates low.

Each of these definitions has been studied in part. The computational complexity of common cryptographic algorithms is well-known [40, 44, 20, 26, 35, 16, 23, 43], as are the lower bounds on routing hops necessary to achieve various security properties [51]. Concrete systems such as distributed hash table (DHT) overlay networks have been well-researched, and much design effort [41, 48] has been expended to make them scalable to a large number of nodes while while keeping network communcation low [14, 52]. To date, however, there has been surprisingly little attention paid to the fundamental *philosophical* issues associated with implementing security solutions as systems grow in size, with a few notable exceptions[2]. Further, there has been just as little work towards aggregating these inherent scalability issues underlying the various subfields of security. To the best of our knowledge, no work has attempted to reconcile these issues into a cohesive whole. Indeed, these possible limits to the scope of computer security as a whole have thus far seen little deliberation. Thus, the purpose of this paper is to consider these limits, and explore whether or not there indeed exist cardinal upper bounds on the effectiveness of computer security.

Note that for the remainder of this paper, the term *security* will be taken to mean some subset of *confidentiality*, *integrity*, *availability* (*CIA*), but not necessarily all three. While recently, *privacy* has received increasing attention and has been distinguished from *confidentiality*, privacy is outside the scope of this paper and security will be the classic *CIA* definition.

1.3 Large-Scale Software Systems

Very-large-scale software systems differ from those of smaller scale in fundamental and unavoidable ways. Due to their scale, issues that are not appreciable in smaller systems can become problems as systems grow in scope. Apart from their sheer size (size of code base, number of users, amount of data, etc.), ultra-large-scale systems can be characterized by certain dimensions, some (but not all) of which may be shared by smaller systems. Northrop et al. [27] suggest the following characterization in their extensive 2006 study of large-scale systems:

1. *Decentralization* - decentralization of data, development, and operation.

2. *Dynamic system requirements* - changing requirements that are varied, uncertain, and often conflicting with one another.

[2]For instance, [4]

3. *Continuous and dynamic evolution* - constant and continuous additions and removals of (and changes to) system components, performed online without the system going offline

4. *Commonplace failures* - hardware and software failures are a fact of life in ultra-large systems

5. *New methods of control* - new methods of operation of the system are necessary for large-scale systems

6. Humans will be more than users of the system, they will be part of the system

These characterizations all have implications for scalable security design and their interactions with issues and current practice will affect the dicsussions throughout the rest of the paper. These dimensions can be seen as underlying assumptions of the very-large system. Although we may not mention all of these explicitly, the reader is encouraged to keep these dimensions in mind while reading the remainder of the paper.

The rest of the paper is organized as follows. Section 2 overviews the issues with software failures due to security vulnerabilities at a high level. Sections 3, 4, and 5 consider more specific and low-level issues associated with very-large-scale systems. Specifically, section 3 discusses the viability of scaling intrusion detection to large systems. Section 4 visits the underlying cryptographic primitives on which much of modern security is constructed. Section 5 provides discourse on issues of decentralization and distribution of computational resources. Section 6 mentions work that is similar in spirit to the present paper. Finally, section 7 serves as the conclusion and provides a view to the future, with some direction for further work.

2. SOFTWARE FAILURES

It is clear that as a software system grows in size and complexity, the probability of software bugs being present increases as well. Software failures are a fact of life in large systems [27]. When these programming errors take the form of exploitable security-vulnerability bugs, parts of, or even the whole system, may be compromised. Since the root cause of these bugs is the inability of humans to write perfect software, it is impossible to mitigate failures completely. Instead, while work is being done in an attempt to prevent some of these bugs [36], most solutions explore techniques for limiting the damage caused by these human failures rather than prevention.

Starting from this assumption of errors being present *somewhere* in the system, robustness of other subsystems - their components, and their tolerance to faults, become of paramount importance in a security context. High-level security goals can be applied directly as techniques to improve system robustness.

For instance, Provos et al. present a method of enforcing **separation-of-duty** by attempting to prevent privilege escalation [39]. Indeed, implementing and enforcing access control policies can limit damage to a system as a result of compromise, for a given attack vector.

Furthermore, common software engineering practices that allow for high system scalability may help with reducing damage from attacks as well. It is common security practice to ensure that systems are loosely-coupled with respect to security; for instance, in a secure password-based authentication protocol, a compromised password should not compromise past sessions, and vice-versa [51]. By reducing interdependence among components of a system, some attacks are unable to propagate to necessary points in their attack path, rendering the attack less harmful.

3. INTRUSION DETECTION

A common technique that has been enjoying continuing popularity increases is the employ of network intrusion detection and intrusion prevention mechanisms in networked systems. However, current intrusion detection systems (IDSs) would be unable to cope with the demands of ultra-large-scale systems. However, the issue is more fundamental than a lack of the necessary technology; there are intrinsic problems regarding the scope of IDSs. One potential limit is the timeliness of detection; whether an IDS is capable of providing timely alerts (ideally, real-time). Although this is a significant concern for intrusion detection, this is beyond the reach of the paper and we do not discuss it further. Below we present a discussion of false alarm rate and related issues, and their interaction with the potential of scalability for intrusion detection systems.

3.1 False Alarms

In [4], Axelsson presents a Bayesian model to analyze and evaluate the theoretical limits of intrusion detection systems in the presence of false positives and false negatives. Axelsson concludes that it is an unrealistic goal to have a low false positive rate while still being effective in detecting malicious activity. He measures the effectiveness of an IDS in terms of usability for the user tasked with investigating the alarms triggered by the IDS. Specifically, Axelsson uses as a metric the Bayesian detection rate [3]. If this value is not reasonably high (on the order of 30% or more), the administrator will be unable to treat alarms raised by the system seriously, and will begin to ignore them. Axelsson measures how this value depends on the interaction between detection rate[4] and false alarm (false positive) rate. As an illustration, the paper cites that a false alarm rate of 10^{-5}, or even several orders of magnitude lower, is necessary in order to reach psychologically-acceptable Bayesian detection rates while maintaining the technical ability (detection rate) to catch intrusions. In the current landscape[5] of the IDS market, false positive rates are on the order of 10^{-2} or worse [6]. These relatively high false positive rates are often attributed to lack of understanding of the nature of intrusion events [18], indicating that further research may take multiple paths, including classification of intrusions and the engineering of better IDSs.

In relation to the scalability of an IDS (that is, its ability to be effective for an ultra-large system), it may be necessary for organizations providing intrusion detection services to resort to hiring employees whose jobs consist of checking each alarm and manually classifying false alarms from true ones. It is likely that the number of such employees to hire increases linearly with the size of the system to protect, undermining the potential scalability of intrusion detection systems. While [4] emphasizes that we cannot yet *conclude* that the necessary precision for an IDS cannot be reached, it is clear that we have a long way to go before this can be realized.

4. CRYPTOGRAPHY

As much of today's security depends on cryptographic primitives, special mention must be reserved for the potential scalability of these operations.

The computational complexity of popular cryptographic algorithms have been thoroughly studied for both public-key [40, 44, 20, 26, 35, 16, 23, 22] and symmetric-key systems [43]. Although new algorithms are likely to appear [24], current trends show that new methods are often derived from the old ones. This is evident in examples such as 3-DES being derived from DES, Twofish from Blowfish, and the ElGamal system being based on the Diffie-Hellman system [16]. Thus, the current systems are likely to continue to be relevant, at least indirectly. Hence, at least for the near future it is a useful exercise to consider scalability as limited by today's cryptographic algorithms.

4.1 Public Key Cryptography

It is commonly accepted that current public-key methods are too slow to be used for encryption and decryption of large messages directly. Popular public-key methods such as RSA [40], Diffie-Hellman [15], elliptic curve cryptography [35, 26], and the ElGamal cryptosystem [16] used with various underlying algorithms are indeed too slow to be used directly. To see this, one must only look glance the large body of literature[6] dealing with minute optimizations of these cryptosystems, each targetted at specific architectures. While this is not entirely true, and public-key methods (such as the McEliece [32] and NTRU [22] systems) exist that are relatively fast, these cryptosystems suffer from other problems and in practice are discarded in favour of the more popular, but slower ones.

However, as systems like McEliece's and NTRU demonstrate, there is no fundamental reason[7] for public-key cryptography to be too inefficient to scale their encryption and decryption algorithms to large messages. Thus, it is entirely possible that we will witness the emergence of more efficient methods to replace the current ones. There is also a considerable amount of continued work on improving the speed of current methods. For instance, Aboud et al. [2] present new and faster algorithms for achieving encryption and decryption in the RSA scheme, touting speedups of multiple

[3] the probability of an event being an intrusion given that the IDS triggers an alarm

[4] The detection rate is the probability of the IDS correctly raising an alarm, given that an intrusion *is* actually happening (contrast this to the false alarm rate, which is the probability that an alarm is raised given that an intrusion *is not* happening)

[5] We were unable to find more recent data in published papers, so this represents the landscape in 2006. We are almost certain, however, that the order of magnitude at the time of the present paper (2012) is the same.

[6] For instance, [42] or [34]

[7] Or rather, one hasn't yet been shown

orders of magnitude. Furthermore, even with the status quo there is the commonly-used workaround of encrypting and sharing a symmetric key using the public-key method, then performing subsequent operations with symmetric cryptography using that key. With this method, the scalability of public-key cryptography becomes entirely dependent on key exchange, and on the symmetric-cryptography algorithms used. Taking this view, the scalability limits of public-key cryptography can be no worse than those of symmetric cryptography together with key exchange, which follow to be discussed in sections §4.2 and §4.3, respectively.

Finally, a brief note must be made regarding management of public keys. Discourse on public key infrastructure (PKI) is beyond the scope of this paper, but effective scaling of public-key cryptography in very-large systems may depend on the scalability, and perhaps more importantly, the trust, of the underlying key management system. The security and scalability issues of PKI have been thoroughly studied, with many papers and articles published on the subject [17]. Whether or not some variant of PKI will be a useful mechanism in the era of ultra-large systems remains to be seen, and key management continues to be an area of open research.

4.2 Symmetric Key Cryptography

Symmetric key cryptography is typically used in tandem with public-key cryptography [47], yet it is the symmetric-key algorithms that are the backbone of a cryptosystem, encrypting and decrypting the majority of the data. Therefore, computational efficiency is necessarily at the heart of a symmetric key method. Characteristic examples of such algorithms include AES (Rijndael) [1], 3-DES (a more secure variant of DES) [5], and Twofish (a more secure variant of Blowfish) [43]. In practice, even at the current scale such algorithms are often limiting factors in terms of computational cost. As with public-key methods, one must only look at the wealth of the body of work[8] dealing with specialized versions of these algorithms for different architectures, and platform- or hardware-specific optimizations. It is possible that there exist information-theoretic upper bounds on transmission of keys and secret information. Similar *lower bounds* for symmetric cryptography have been shown [3], but to the best of our knowledge an upper bound has not yet been demonstrated. Therefore, it is reasonable to claim that there are (as of yet) no fundamental limits on the complexity of symmetric key cryptography, and thus no fundamental limits on their potential scalability[9].

4.3 Key Exchange

A closely-related problem to the complexity of the cryptographic algorithm primitives is key exchange over unauthenticated networks. Apart from the computational complexity discussed in the preceding sections, schemes for key exchange also face other challenges. The network latency induced with key exchange protocols represents a critical lower bound on the time to perform key exchange. Thus, in large-scale systems, this can lead to limits on the scope of the system in the case when it is necessary to perform key exchanges quickly between multiple ephemeral authenticating parties [10] that are separated by large geographical distances. In the absence of efficient mechanisms for exchange among multiple authenticating parties, latency can undermine the scalability of a system.

However, over the past two decades there has been much advanacement in efficient multiple-peer key exchange. A 2003 paper by Katz and Yung [25] describes an efficient protocol for asymmetric key exchange in a group, achieving $O(1)$ in the number of authentication rounds. Previous work [10, 9] had required $O(n)$ in the number of rounds. Still, to achieve $O(1)$ message rounds, the system presented in [25] trades off local computation, requiring $O(n)$ signature checks by each group member. Since the paper by Katz and Yung, there have been numerous other schemes that involve a different trade-off point, such as the one presented in [11], which requires $O(\log n)$ in both the number of rounds and the number of signature checks.

Notwithstanding, these existing schemes resemble each other as they are all either based directly on the Diffie-Hellman problem (DHP), or involve a transformation from a different scheme into an instance of the DHP. If the DHP is ever shown not to be hard to break, these schemes could no longer be used. Indeed, as discussed in the preceding sections, it is not yet clear which (if any) of today's current ciphers will be used for the large-scale systems of tomorrow. Thus, unless efficient schemes not based on the Diffie-Hellman problem are discovered, it is possible that group key exchange may face future scalability challenges.

5. DECENTRALIZATION AND NETWORK ISSUES

As the large systems of tomorrow will necessarily be decentralized, distributed systems represent the landscape of the ultra-large system (if they do not already). Both the number and scope of systems such as peer-to-peer networks and cloud computing platforms have witnessed steady growth since their introduction. Data centres are increasingly distributed, leading to an ever-greater demand for protection from network intruders. Indeed, due to the potentially large scale of certain distributed systems, the field is intrinsically linked to the quest for scalable security solutions. Further, prevention of attacks in large networked systems is notoriously difficult. Distributed denial-of-service (DDoS) attacks are known to be simple to perform and scale well for the attacker. Although a great deal of effort has gone into protection against DoS and DDoS attacks, the problem remains an open challenge. Proposed defense mechanisms can often be effective in special cases[11], but even then their scalability in relation to the scalability of the attack is questionable. Indeed, prevention methods may not scale as well as the attacks do; thus, for large-scale attacks on very-large systems, DoS attacks may become unfeasible to defend against due to these potential scaling discrepancies. DoS prevention is implicitly discussed in the context of intrusion detection (§3), but further discourse is beyond the reach of this paper.

[8]For example, [33, 31]

[9]Actually, there are network issues, which are discussed in §4.3 in the context of key exchange

[10]These can be users, or components of a system requiring temporary keys, for example

[11]for example, [29] presents a solution in the special case of running in a bandwidth-underprovisioned public cloud

Moreover, there are known issues with the security of the fundamental distributed systems topic of group and multicast communication. As very-large-scale systems will be decentralized, it is likely that such communication paradigms will play a large role. Previous work has discussed the problem of securing such forms of communication, and outlined the difficulties of making such security efficient [13, 12]. Important applications include communication in distributed hash table (DHT) network overlays, whose security and scalability considerations we delegate to works such as [50, 46], apart from an implicit mention under the umbrella of file systems in §5.1.2. We do not provide further discussion of this vast field, but note that it is a rich area of study with much research opportunity.

In the following subsection we overview scalability and security considerations of another fundamental property of large distributed software systems: that of distributed storage. We find that although their underlying issues are not ones setting fundamental limits on scalable secure storage, they provide a taste of the present challenges of scalable secure decentralized systems and illustrate that there is still much work to be done in this area.

5.1 Data Storage

As we advance further into the era of Big Data, storage demands continue to see increases. Indeed, for very large systems, storage of data becomes perhaps an even greater issue than for smaller-scale systems. Of the characterization metrics proposed in section §1.3, particularly relevant ones for storage are *decentralization* and *commonplace failures*. These phenomena can even be seen in today's larger systems, with geographically-separated server farms being commonly used, and resilience and fault-tolerance being a requirement of data storage. Although in general, security of large-scale storage is intrinsically linked to security of distributed systems, it is worth mentioning current problems unique to storage, that require solutions before we witness their scaling to extremely large sizes.

5.1.1 Databases

As databases (whether they be traditional SQL relational databases, or the increasingly-popular so-called *NoSQL* alternatives) contain ever-growing amounts of data, the total value of their stored information increases as well. In turn, this makes databases attractive targets for attackers. Therefore, providing confidentiality, integrity, and availability for the data must also increase in importance. While the field of relational database security is rich with decades of extensive study and innovation, scalability-oriented NoSQL alternatives have not been scrutinized as carefully thus far. Okman et al. present a survey of security issues in NoSQL databases [38], focusing on *Cassandra* and *MongoDB*, the two most popular NoSQL stores at the time of writing. Okman et al. conclude that these datastores are lacking in security features. There is irony in the fact that they are designed for high availability and aim to scale to huge data, yet the security they lack is perhaps most important precisely at the intended very-large scale.

5.1.2 File Systems

Large, distributed file systems represent an important example of large-scale storage, and securing such a file system presents unique challenges. Distributed file systems form a mature field with a large body of research and countless different implementations of such systems. This field is perhaps one of the best-developed in terms of offering true scalability. However, the security of these systems is questionable.

In one sense, distributed file systems have already proven to scale to ultra-large systems. In [19], Ghemawat et al. present the design and implementation of the Google File System (GFS), used internally by Google for large-scale storage needs. The authors cite hundreds of terabytes across thousands of disks for one of the larger clusters, as far back as 2003. Since then, storage requirements have grown significantly; hundreds of terabytes per-cluster is no longer a colossal amount of storage. Still, the GFS has been significant in shaping newer systems. The GFS provided the inspiration for the open-source Hadoop Distributed File System (HDFS) [7], by which it has been largely superseded as the de facto very-large filesystem of choice. As of 2010, Yahoo! uses HDFS to manage 25 petabytes of data across 25 000 servers, with the largest cluster consisting of 3,500 nodes [45].

Clearly, thus far distributed file systems have been able to scale very effectively, in part due to extensive use of multiple caching levels [49]. Whether caching and other techniques will be able to handle the ultra-large data of the future remains to be seen, but we expect this trend to continue at least into the near future. Even if they are indeed scalable, the security of these file systems of today's largest file systems is questionable. Although authentication via access control methods is one of the primary goals of many file systems, both GFS and HDFS assume that all nodes in the system are trusted [49]. Thus, security of these systems depends entirely upon the security of the individual nodes, and that of their underlying network. The difficulty of preventing attacks as systems scale up has been noted in previous sections, and the probability that there exist compromised nodes increases as the file system's size increases. Similarly, prevention of attacks against the underlying communication network suffers the same problems as previously mentioned (for instance, intrusion detection). Thus, we believe that this model is not sustainable for ultra-large-scale systems and that distributed file system design must undergo a paradigm shift toward the assumption that some nodes may be untrusted.

There is hope, however, and research is actively being conducted towards this goal. For instance, in a 2007 paper by Leung et al. [28], the authors note the difficulty of providing authentication and authorization in large distributed file systems containing sensitive user data. The authors cite the inadequacies of previous solutions, and go on to design and describe a system that is able to scale effectively to petascale file systems while remaining more secure than other contemporary systems.

6. RELATED WORK

To date, there has not been a thorough consideration of the security of very-large-scale systems, nor one exploring the fundamental scalability of aggregated security techniques. Northrop et al. [27] discuss major research areas related to ultra-large-scale systems at a scope far beyond that of this

paper, but only briefly touch upon the security of such systems. Their treatment of security is at a very high level and they only consider security as a quality attribute of a system, choosing not to explore the associated scalability issues in depth. Axelsson presents a philosophical and probabilistic treatment of limits to the scope of intrusion detection [4], citing the *base-rate fallacy* as a fundamental problem potentially limiting the scalability of intrusion detection. However, the scope is constrained to the comparatively narrow topic of intrusion detection, and further areas are not explored. To the best of our knowledge, ours is the first attempt at a broad treatment of the scalability of general computer security. Although a more thorough survey would be salient, we have provided a step towards understanding the scalability of security as a whole.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have considered scalable security, and investigated whether there exist fundamental limits to the scope of secure system design at the ultra-large scale. We have surveyed the scalability of various subfields of computer security and have attempted to combine the underlying issues in these subfields into a single view of security-scalability. We have discovered that, although there are some potentially unsurmountable limits undermining some subfields, on the whole the future does not seem as bleak as we had expected at the outset. We have noted that underlying all large-scale systems are inevitable vulnerabilities that potentially undermine all aspects of a system's security. We have found that, while *prevention* of vulnerabilities is a noble goal, vulnerability-free large-scale software is unattainable. Therefore, we recommend that research move in the direction of *minimizing* the damage caused by vulnerabilities, rather than attempting to reduce this damage to zero. Further, we considered the task of detecting and preventing attacks (intrusions) on such systems, finding that intrusion detection is limited by the rate of false positives. Although false positive rates must be on the order of 10^{-5} or lower in order to be psychologically acceptable to network administrators, there does not appear to be any unbreakable law stating that in the future intrusion detection cannot become this precise. However, we believe that we still far away from this sort of precision, if it is indeed ever reached. We find that for underlying cryptographic algorithms, the situation is less pessimistic. Given the current public knowledge, it does not seem as though there are any fundamental limits on the computational complexity of both public-key and symmetric-key methods. Instead, we find the more significant issue to be the network latency required for authentication – for the generation and sharing of ephemeral keys. Latency represents an absolute limit[12] tied to the speed of light, and this issue can contribute to limit potential scalability. Finally, we look at issues in the distributed nature of large-scale systems, with a focus on storage issues. We find that, while the most attractive scalable databases (NoSQL solutions) lack in security features, they are still immature and there is no reason that they cannot be made more secure in the future. Similarly, highly-scalable distributed file systems exist, but are not inherently secure. Instead, they delegate security to the underlying nodes and the security of their internal network topology. We believe that this is a potential failure point as systems scale up, but this can be remedied by revising the assumptions of a trusted network, implementing future solutions in light of these issues. Here, again, we do not find unsurmountable hurdles.

[12]The absoluteness of the speed of light has been disputed, but here we take the view that it is indeed absolute, given current beliefs.

Finally, we provide direction for future work. We believe that treatments with the same philosophy as Axelsson's treatment of intrusion detection systems [4] are worthwhile. In particular, a formal treatment of the limits of cyptography in the framework of information theory would be prudent. Similarly, a formal definition of damages due to vulnerabilities may provide insight into the acceptable ratios of software failures (vulnerabilities) to system size, or may conclude that theoretically, the inevitable presence of these failures can lead to full system compromise. To that end, we advocate the current mentality of limiting damages as a result of vulnerabilities, rather than preventing damages entirely. We call for further steps in the direction of secure, scalable distributed systems design, noting that while much work has been done [28], there is even more left to do. We note, however, that these are challenging areas of research and do not expect especially rapid progress. As a concluding thought, we solicit research into the emerging field of *security economics*. Some of the underlying issues discussed in this paper may face stout challenges in attempts to be explained by technical or formal means. However, the framework of economics may be able to more easily explain, and provide solutions to, these issues; if attackers have no incentives, does a system really need a whole host of protection mechanisms?

Although there do exist some fundamental issues in the quest for scaling security to enormous systems, we move into the era of ultra-large-scale with hope and optimism.

8. REFERENCES

[1] Specification for the advanced encryption standard (aes). Federal Information Processing Standards Publication 197, 2001.

[2] S. Aboud, M. Al-Fayoumi, M. Al-Fayoumi, and H. Jabbar. An efficient rsa public key encryption scheme. In *Information Technology: New Generations, 2008. ITNG 2008. Fifth International Conference on*, pages 127 –130, april 2008.

[3] R. Ahlswede and I. Csiszar. Common randomness in information theory and cryptography. i. secret sharing. *Information Theory, IEEE Transactions on*, 39(4):1121 –1132, jul 1993.

[4] S. Axelsson. The base-rate fallacy and the difficulty of intrusion detection. *ACM Trans. Inf. Syst. Secur.*, 3:186–205, August 2000.

[5] W. C. Barker, N. I. of Standards, and T. (U.S.). *Recommendation for the Triple Data Encryption Algorithm (TDEA) block cipher [electronic resource] / William C. Barker*. U.S. Dept. of Commerce, Technology Administration, National Institute of Standards and Technology, Gaithersburg, MD :, 2004.

[6] D. Bolzoni, S. Etalle, and P. Hartel. Poseidon: a 2-tier anomaly-based network intrusion detection system. In *Information Assurance, 2006. IWIA 2006. Fourth IEEE International Workshop on*, pages 10 pp. –156,

april 2006.

[7] D. Borthakur. *The Hadoop Distributed File System: Architecture and Design*. The Apache Software Foundation, 2007.

[8] J. Bosque and L. Perez. Theoretical scalability analysis for heterogeneous clusters. In *Cluster Computing and the Grid, 2004. CCGrid 2004. IEEE International Symposium on*, pages 285 – 292, april 2004.

[9] E. Bresson, O. Chevassut, and D. Pointcheval. Dynamic group diffie-hellman key exchange under standard assumptions. In *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques: Advances in Cryptology*, EUROCRYPT '02, pages 321–336, London, UK, UK, 2002. Springer-Verlag.

[10] E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater. Provably authenticated group diffie-hellman key exchange. In *Proceedings of the 8th ACM conference on Computer and Communications Security*, CCS '01, pages 255–264, New York, NY, USA, 2001. ACM.

[11] M. Burmester and Y. Desmedt. A secure and scalable group key exchange system. *Information Processing Letters*, 94(3):137 – 143, 2005.

[12] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas. Multicast security: a taxonomy and some efficient constructions. In *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 708 –716 vol.2, mar 1999.

[13] G. Caronni, K. Waldvogel, D. Sun, and B. Plattner. Efficient security for large and dynamic multicast groups. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 1998. (WET ICE '98) Proceedings., Seventh IEEE International Workshops on*, pages 376 –383, jun 1998.

[14] M. Castro, P. Druschel, Y. Charlie, and H. A. Rowstron. Exploiting network proximity in peer-to-peer overlay networks. Technical report, 2002.

[15] W. Diffie and M. Hellman. New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644 – 654, nov 1976.

[16] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. Blakley and D. Chaum, editors, *Advances in Cryptology*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18. Springer Berlin / Heidelberg, 1985. 10.1007/3-540-39568-7_2.

[17] C. Ellison and B. Schneier. Ten risks of pki: What you're not being told about public key infrastructure. *Computer Security Journal*, 16(1):1–7, 2000.

[18] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Macia-Fernandez, and E. Vazquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, 28(1âĂŞ2):18 – 28, 2009.

[19] S. Ghemawat, H. Gobioff, and S.-T. Leung. The google file system. *SIGOPS Oper. Syst. Rev.*, 37(5):29–43, Oct. 2003.

[20] N. Gura, A. Patel, A. Wander, H. Eberle, and S. C. Shantz. Comparing elliptic curve cryptography and rsa on 8-bit cpus. In M. Joye and J.-J. Quisquater, editors, *Cryptographic Hardware and Embedded Systems - CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 925–943. Springer Berlin / Heidelberg, 2004. 10.1007/978-3-540-28632-5_9.

[21] M. D. Hill. What is scalability? *SIGARCH Comput. Archit. News*, 18(4):18–21, Dec. 1990.

[22] J. Hoffstein, J. Pipher, and J. Silverman. Ntru: A ring-based public key cryptosystem. In J. Buhler, editor, *Algorithmic Number Theory*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer Berlin / Heidelberg, 1998. 10.1007/BFb0054868.

[23] M.-S. Hwang, C.-C. Chang, and K.-F. Hwang. An elgamal-like cryptosystem for enciphering large messages. *Knowledge and Data Engineering, IEEE Transactions on*, 14(2):445 –446, mar/apr 2002.

[24] T. Jamil. The rijndael algorithm. *Potentials, IEEE*, 23(2):36 – 38, april-may 2004.

[25] J. Katz and M. Yung. Scalable protocols for authenticated group key exchange. In D. Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 110–125. Springer Berlin / Heidelberg, 2003.

[26] N. Koblitz, A. Menezes, and S. Vanstone. The state of elliptic curve cryptography. *Designs, Codes and Cryptography*, 19:173–193, 2000. 10.1023/A:1008354106356.

[27] R. P. G. J. G. R. L. T. L. e. a. L Northrop, P Feiler. *Ultra-Large-Scale Systems*. 2006.

[28] A. W. Leung, E. L. Miller, and S. Jones. Scalable security for petascale parallel file systems. In *Proceedings of the 2007 ACM/IEEE conference on Supercomputing*, SC '07, pages 16:1–16:12, New York, NY, USA, 2007. ACM.

[29] H. Liu. A new form of dos attack in a cloud and its avoidance mechanism. In *Proceedings of the 2010 ACM workshop on Cloud computing security workshop*, CCSW '10, pages 65–76, New York, NY, USA, 2010. ACM.

[30] E. Luke. Defining and measuring scalability. In *Scalable Parallel Libraries Conference, 1993., Proceedings of the*, pages 183 –186, oct 1993.

[31] S. Manavski. Cuda compatible gpu as an efficient hardware accelerator for aes cryptography. In *Signal Processing and Communications, 2007. ICSPC 2007. IEEE International Conference on*, pages 65 –68, nov. 2007.

[32] R. J. Mceliece. A public-key cryptosystem based on algebraic coding theory. *DSN progress report*, 42(44):114–116, 1978.

[33] M. McLoone and J. McCanny. A high performance fpga implementation of des. In *Signal Processing Systems, 2000. SiPS 2000. 2000 IEEE Workshop on*, pages 374 –383, 2000.

[34] E. Michalski and D. Buell. A scalable architecture for rsa cryptography on large fpgas. In *Field Programmable Logic and Applications, 2006. FPL '06. International Conference on*, pages 1 –8, aug. 2006.

[35] V. Miller. Use of elliptic curves in cryptography. In H. Williams, editor, *Advances in Cryptology - CRYPTO '85 Proceedings*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426. Springer

Berlin / Heidelberg, 1986. 10.1007/3-540-39799-X_31.

[36] T. Nakashima, M. Oyama, H. Hisada, and N. Ishii. Analysis of software bug causes and its prevention. *Information and Software Technology*, 41(15):1059 – 1068, 1999.

[37] D. Nussbaum and A. Agarwal. Scalability of parallel machines. *Commun. ACM*, 34(3):57–61, Mar. 1991.

[38] L. Okman, N. Gal-Oz, Y. Gonen, E. Gudes, and J. Abramov. Security issues in nosql databases. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on*, pages 541 –547, nov. 2011.

[39] N. Provos, M. Friedl, and P. Honeyman. Preventing privilege escalation. In *Proceedings of the 12th conference on USENIX Security Symposium - Volume 12*, SSYM'03, pages 16–16, Berkeley, CA, USA, 2003. USENIX Association.

[40] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21:120–126, February 1978.

[41] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In R. Guerraoui, editor, *Middleware 2001*, volume 2218 of *Lecture Notes in Computer Science*, pages 329–350. Springer Berlin / Heidelberg, 2001. 10.1007/3-540-45518-3_18.

[42] A. Satoh and K. Takano. A scalable dual-field elliptic curve cryptographic processor. *Computers, IEEE Transactions on*, 52(4):449 – 460, april 2003.

[43] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, and N. Ferguson. Twofish: A 128-bit block cipher. In *in First Advanced Encryption Standard (AES) Conference*, 1998.

[44] M. Shand and J. Vuillemin. Fast implementations of rsa cryptography. In *Computer Arithmetic, 1993. Proceedings., 11th Symposium on*, pages 252 –259, jun-2 jul 1993.

[45] K. Shvachko, H. Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1 –10, may 2010.

[46] E. Sit and R. Morris. Security considerations for peer-to-peer distributed hash tables. In P. Druschel, F. Kaashoek, and A. Rowstron, editors, *Peer-to-Peer Systems*, volume 2429 of *Lecture Notes in Computer Science*, pages 261–269. Springer Berlin / Heidelberg, 2002. 10.1007/3-540-45748-8_25.

[47] W. Stallings. *Cryptography and network security: principles and practice*. The William Stallings Books on Computer and Data Communications. Pearson/Prentice Hall, 2006.

[48] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *SIGCOMM Comput. Commun. Rev.*, 31:149–160, August 2001.

[49] T. Thanh, S. Mohan, E. Choi, S. Kim, and P. Kim. A taxonomy and survey on distributed file systems. In *Networked Computing and Advanced Information Management, 2008. NCM '08. Fourth International Conference on*, volume 1, pages 144 –149, sept. 2008.

[50] G. Urdaneta, G. Pierre, and M. V. Steen. A survey of dht security techniques. *ACM Comput. Surv.*, 43(2):8:1–8:49, Feb. 2011.

[51] T. Wu. The secure remote password protocol. In *Internet Society Network and Distributed System Security Symposium*, pages 97–111, March 1998.

[52] H. Zhang, A. Goel, and R. Govindan. Incrementally improving lookup latency in distributed hash table systems. *SIGMETRICS Perform. Eval. Rev.*, 31:114–125, June 2003.